Enhancing Video Error Resilience by Using Data-Embedding Techniques

Wen-Nung Lie, Tom C.-I. Lin, and Chia-Wen Lin

***Abstract*—In this paper, error-resilient video coding schemes based on data-embedding techniques are proposed for the H.263+ codec. Data embedding, popularly applied to secret hiding and digital watermarking, is now used to convey error recovery information to the decoder via a covert channel, without causing significant increase in transmission bitrate. Our embedded information provides implicit macroblock (MB) delimiters for resynchronization in presence of channel errors. In this way, the decoder is capable of isolating erroneous MBs with the extracted information. A set of variational schemes is proposed, extensively analyzed, and compared to the competitive counterparts (e.g., the original H.263+ TMN8 and its synchronization-enhanced version) at the same bitrate. Experimental results show that our data embedding process decreases the average peak signal-to-noise ratio (PSNR) in error-free conditions by 0.3 dB (light data embedding) to 1.8 dB (heavy data embedding), but it is capable of achieving a significant PSNR improvement up to 2 and 9.5 dB when the bit error rate is 10^{-5} and 10^{-3}, respectively. We also provide suggestions of how to adaptively apply the proposed schemes for different channel error conditions and different video contents.**

***Index Terms*—Data embedding, error resilience, H.263+, resynchronization.**

I. Introduction

SEVERAL video compression standards, such as H.26x and MPEG-1/2/4, have been proposed in the past decade for storage and communication purposes. However, as the video data are highly compressed, they become sensitive to errors caused by unreliable transmission channels. Often, error propagation will be accompanying, meaning that an error at any position of the bit stream will disable not only the decoding of the word that contains it, but also the following ones until a synchronization symbol is met. Furthmore, reconstruction error in a single pixel sample will affect all the samples that are directly or indirectly predicted from it, thus leading to video quality degradation. Usually, error concealment techniques such as zero-motion replacement or spatial interpolation [1] are applied to solve this predicament.

Several methods have been proposed to avoid error propagation, such as inserting resynchronization markers [3], reversible variable-length coding (RVLC) [4], and error-resilient entropy coding (EREC) [5]. Inserting resynchronization markers periodically or adaptively is the most simple and effective method for enhancing error resilience, but will introduce lots of redundancies and increase the bit rate rapidly. Each GOB startcode in H.263+ spends at least 31 bits and is obviously an expensive effort. The RVLC scheme, recommended as an option in H.263+, is capable of achieving unique decoding in both the forward and reverse directions of the bit stream. According to related reports [6], [7], RVLC sacrifices 1.5%–12% of coding efficiency for motion vectors (MVs) [6] and DCT coefficients [7], compared to the traditional VLC. Besides, RVLC cannot rescue data in between the first and last errors when more than one error is present in the same packet. On the other hand, EREC, converting the traditional VLC bit stream into fixed-length frames of data, allows the decoder to synchronize bit streams at the start of each EREC frame. The major drawbacks of EREC are no guarantee of frame spatial synchronization and the requirement of highly-protected auxiliary information.

Recently, researchers proposed several error-resilient methods based on data embedding techniques, which are originally proposed for, e.g., watermarking [2] and steganalysis [18]. They applied data embedding schemes to establish another covert channel for transmitting important information that enhances error resilience capability without increasing the bit rate significantly [8]–[12]. In [8], the parity check bits of MV codewords in a frame are embedded into the half-pel MVs of its following frame to recover the corrupted MVs with a single bit error, but degrade the video quality by 0.3–1.0 dB of PSNR. The method of parity embedding error detection (PEED) [9], on the other hand, embeds parity check bits of MB data into MVs and DCT coefficients of the following frame, but leaves the bits between the synchronization code and the first following error bit discarded even thought they are error-free. In [10], a feature vector containing information such as smoothness and edge orientation is computed for each encoded MB and embedded into its companion counterpart. These features are thus extracted at the decoder to facilitate error concealment, especially for the interpolation of intra-coded MBs. In [11], Leou *et al.* proposed to embed codeword indices, generated by the vector quantization (VQ) technique, of an image into the JPEG quantized DCT coefficients by the modulo 2 method. At the receiver, the VQ information is extracted to reconstruct the corrupted blocks in presence of channel errors. In [12], Okada *et al.* proposed to embed the so-called check marker of each MB into its own DCT coefficients. This method provides better error detection capability than traditional MPEG4 decoder, but still has to discard bits between the error bit and next synchronization code.

When the data embedding technique is applied to visual communication, the most concerned problems are: what, where, and how to embed the information. Take the above-mentioned cases for examples:

Manuscript received April 3, 2003; revised June 24, 2004. This work was supported by the National Science Council, Taiwan, R.O.C., under Grant NSC91-2213-E-194-012. This paper was recommended by Associate Editor C. W. Chen.

W.-N. Lie and T. C.-I. Lin are with the Department of Electrical Engineering, National Chung Cheng University, 621 Taiwan, R.O.C. (e-mail: wnlie@ee.ccu.edu.tw).

C.-W. Lin is with the Department of Computer Science and Information Engineering, National Chung Cheng University, 621 Taiwan, R.O.C. (e-mail: cwlin@cs.ccu.edu.tw).

Digital Object Identifier 10.1109/TCSVT.2005.861948

1) what to embed: parity check bits [8], [9], [12] or macroblock features [10], [11];
2) where to embed: MVs [8], [9] or DCT coefficients (in following frame [9], [10], the same frame [11], or even the same MB [12]);
3) how to embed: change of MVs to specific half-pixel positions [8], [9] or reset (with ± 1) or modulo 2 change of DCT coefficients [9]–[12].

Among them, the content of embedded information will be crucial to error resilience capability and should be the first concern. One kind of information that is critical to error recovery but difficult to estimate at the decoder is the resynchronization. Rather than inserting additional resynchronization codes, the number of bits of each encoded MB is embedded. This embedded information is capable of providing implicit delimiters between MBs so that the decoder can skip the corrupted MB data and continue the decoding of next MB. This concept can be applied to standards such as H.263+ and MPEG-2/4 with slight adaptation.

In this paper, we select the H.263+ standard [14] as the implementation platform due to its popular use in visual communication applications, especially in wireless channel environments.

II. Basic Data Embedding Schemes

As addressed in the previous section, MVs and DCT coefficients are the most common places for information embedding. The former has the drawbacks of low embedding capacity (2 bits per MB) and significant quality degradation (since the advantage of half-pixel MVs has been removed). On the other hand, DCT coefficients are advantageous of higher embedding capacity and negligible quality degradation if properly arranged. For the above reasons, we adopt in this paper the strategy of embedding information in DCT coefficients only.

Due to different embedding capacities, we have to adopt different embedding schemes for intra- and inter-coded MBs and different embedded information for I- and P-frames.

A. What to EMBED

For I-frames, we can embed longer information due to their larger embedding capacity. As stated in Section I, data length of an encoded MB (here abbreviated as MB_DL) is helpful in resynchronization if that MB is corrupted due to channel errors. According to experiments, if the quantization scale factor, QP, is larger than 3 in H.263+, the maximal MB_DL will be mostly smaller than 4096 bits. This means that we need to embed 12 bits of MB_DL for each MB for medium-to-low bit-rate transmission.

On the other hand, P-frames have three coding types for MBs: skipped, inter-coded, and intra-coded. Normally, only a few parts are intra-coded. At low bit rates, an inter-coded MB frequently contains a large number of zero blocks, or even the whole MB is skipped and coded with only a single bit "1." If errors occur in the "skipped" run, they will be also propagated to the decoding of nonskipped MBs that immediately follow. Hence, recording the run length of "skipped MBs" is helpful to leap over them and keep on decoding the following nonskipped MBs. There are three possible relations between two nonskipped MBs (denoted as MB_*before* and MB_*after* in order) separated by a run of skipped MBs of length d: 1) located at the same GOB row; 2) located at adjacent GOB rows; and 3) separated by at least one blank GOB (i.e., GOB of all skipped MBs). For cases 2) and 3), the binary code of d may need 7 bits to account for a maximal run length of 99 for the QCIF format. Considering limited embedding capacity for inter-coded MBs in P-frames, location of MB_*after* in a GOB row (denoted as MB_HL) is instead embedded into MB_*before* for decoder use. In this manner, 4-bit information is enough for QCIF images where each GOB row contains 11 MBs. With this information (MB_HL), the decoder can check the position of the first decoded nonskipped MB (which starts with a bit "0") after MB_*before* to see if any error occurs therein.

In addition to protecting errors in skipped MBs, errors occurring in the nonskipped MB bit stream should also be considered. Considering again MB_*after*, its data length can be recorded and embedded into MB_*before*. Eight-bit information (denoted as MB_DL_8) is chosen due to shorter data lengths for most inter-coded MBs in P-frames. This is obviously insufficient if MB_*after* is intra-coded. However, a tradeoff between error resilience and embedding capacity for an inter-coded MB_*before* should be made. Combining the 4-bit MB_HL and 8-bit MB_DL_8 information, a 12-bit datum can thus be embedded so as to identify the start and end bit positions of MB_*after*. Here, a mechanism is designed to notify the decoder of the situation of deficient MB_DL_8. MB_DL_8 is assigned with "00 000 000" to indicate the over-length (i.e., larger than 255) of MB_*after* bit stream. In this case, the decoder should give up re-synchronizing the MB_*after* if it is in errors.

B. Where and How to Embed

Data embedding often causes image quality degradation [15], [16]. The quality degradation of a frame may result in an increase of prediction residues in successive following P-frames, thus further increasing the coding bit rate. For this reason, our proposed scheme considers embedding with tradeoffs between the coding performance (i.e., quality/bit rate) and the error resilience that it can achieve in case of errors.

Now we face the problems of "where" and "how" to embed information so as to cause less influence on quality and bit rate. Basically, information is embedded in DCT coefficients of MBs. Those MBs selected to embed data are called the hosts. In H.263+, all MBs in I-frames are intra-coded and can be selected as the hosts. On the other hand, host MBs in P-frames should be conditionally selected (reject the skipped MBs and those contain no DCT coefficients).

In H.263+, the dc component of each intra-coded MB is coded in a fixed length (8 b). Therefore, embedding (e.g., by the common LSB (least significant bit) method) at dc coefficients does not cause any bit-rate increase, but possibly severe quality degradation. For inter-coded MBs, all DCT coefficients are variable-length-coded and coefficients at lower frequencies should be chosen to avoid significant bit-rate increase. Due to this difference in coding principles, we develop separate embedding schemes for intra- and inter-coded MBs as follows.

- Scheme 1: Embedding data into the two LSBs of a quantized dc coefficient by the direct replacement method.

- Scheme 2: Embedding data into the selected ac coefficients (e.g., the first and second coefficients in the zigzag scanning order) by the modulo 2 method.

In H.263+ standard, dc coefficients of intra-coded MBs are always quantized with a step size $Q_{\rm dc} = 8$. Hence, scheme 1 introduces errors in the range of $-3Q_{\rm dc}- + 3Q_{\rm dc}$. To reduce image degradation at the receiver side, the decoder can always assign the last two bits of each decoded dc coefficient to be "01" after data extraction. In this manner, the dc reconstruction error is narrowed down to be between $(-2Q_{\rm dc}, Q_{\rm dc})$. The same assignment procedure is consistently performed for the local decoder (before the inverse quantization process) at the encoder side. Since our local decoder has taken the embedding effect into consideration, there will be no drift errors in following frames (but the total bit rate is increased slightly). For each intra-coded MB, there are four luminance (Y) and two chrominance (Cb, Cr) blocks, achieving an embedding capacity of 12 b.

On the other hand, embedding in ac coefficients (scheme 2) is considered for inter-coded MBs of P-frames. The following modulo 2 scheme is adopted:

$$\begin{aligned}&\text{If } (C_j \bmod 2 = \text{embedded\_bit}) \text{ then}\\ &\quad C_j \text{ remains unchanged}\\ &\text{Else } C_j = C_j - 1 \text{ (for } C_j \geqq 0) \text{ or}\\ &\quad C_j = C_j + 1 \text{ (for } C_j < 0)\end{aligned}$$

where C_j is the quantized ac coefficient selected for embedding. This scheme is similar to the LSB-replacement method, but tends to get a smaller magnitude after modification, which generally has a shorter VLC code.

To have a comparison of quality degradation between dc and ac embedding, we have the following derivations. The following equation shows the formula of 8×8 inverse DCT (IDCT):

$$\begin{aligned}f(x,y) = &\sum_{u=0}^{7}\sum_{v=0}^{7}\alpha(u)\alpha(v)F(u,v)\\ &\cdot\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right] \qquad (1)\end{aligned}$$

$$\alpha(\lambda) = \begin{cases}\sqrt{\frac{1}{8}}, & \text{for } \lambda = 0\\ \frac{1}{2}, & \text{otherwise}\end{cases} \qquad (2)$$

where $F(u,v)$ represents the de-quantized DCT coefficient and $f(x,y)$ represents a pixel in the image domain. After data embedding, $F(u,v)$ is changed to be $\widetilde{F}(u,v)$. Since DCT is a linear transformation, the noise term $\Delta F(u,v) = \widetilde{F}(u,v) - F(u,v)$ can be separated out from IDCT of $\widetilde{F}(u,v)$ to result in

$$\begin{aligned}\hat{f}_{\rm dc}(x,y) = &\sum_{u=0}^{7}\sum_{v=0}^{7}\alpha(u)\alpha(v)\widetilde{F}(u,v)\\ &\cdot\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]\\ = &f(x,y) + \alpha(0)\alpha(0)\Delta F(0,0) \qquad (3)\end{aligned}$$

$$\begin{aligned}\hat{f}_{\rm ac}^{m,n}(x,y) = &\sum_{u=0}^{7}\sum_{v=0}^{7}\alpha(u)\alpha(v)\widetilde{F}(u,v)\\ &\cdot\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]\\ = &f(x,y) + \alpha(m)\alpha(n)\Delta F(m,n)\\ &\cdot\cos\left[\frac{(2x+1)m\pi}{16}\right]\cos\left[\frac{(2y+1)n\pi}{16}\right] \qquad (4)\end{aligned}$$

where we have assumed dc and ac (at a single frequency (m,n)) embedding in (3) and and (4), respectively.

Let's have an insight into the reconstruction noise terms, i.e., $\Delta f(x,y) = \hat{f}(x,y) - f(x,y)$, to see how they behave with dc and ac embedding. Since the quantization step sizes for dc and ac coefficients are 8 and $2 \cdot QP$, respectively, in H.263+, we have

$$\Delta F(0,0) = 8\delta_{\rm dc} \qquad (5)$$

$$\Delta F(m,n) = 2 \cdot QP \cdot \delta_{\rm ac} \qquad (6)$$

where $\delta_{\rm dc}$ and $\delta_{\rm ac}$ are reconstruction errors corresponding to 2-bit dc and 1-bit ac embedding, respectively.

Hence, we yield

$$\begin{aligned}\Delta f_{\rm dc}(x,y) = &\alpha(0)\alpha(0)\Delta F(0,0)\\ = &\frac{1}{8}\cdot 8\cdot\delta_{\rm dc} = \delta_{\rm dc} \qquad (7)\end{aligned}$$

$$\begin{aligned}\Delta f_{\rm ac}^{m,n}(x,y) = &\alpha(m)\alpha(n)\Delta F(m,n)\\ &\cdot\cos\left[\frac{(2x+1)m\pi}{16}\right]\cos\left[\frac{(2y+1)n\pi}{16}\right]\\ = &k\cdot QP\cdot\left\{\delta_{\rm ac}\cdot\cos\left[\frac{(2x+1)m\pi}{16}\right]\right.\\ &\left.\cdot\cos\left[\frac{(2y+1)n\pi}{16}\right]\right\} \qquad (8)\end{aligned}$$

where k is equal to $(1/(2\sqrt{2}))$ or 1/2, depending on the values of m and n (i.e., the position of the modified coefficient). According to the prior statements, 2-bit embedding in dc coefficient leads to a reconstruction error $-2 \leq \delta_{\rm dc} \leq 1$, while 1-bit embedding in ac coefficient leads to $\delta_{\rm ac} = \pm 1$. We derive the range of pixel reconstruction error, based on the same embedded amount of data (i.e., 2 b), to be

$$-2 \leq \Delta f_{\rm dc}(x,y) \leq 1 \qquad (9)$$

$$\begin{aligned}-1.0\cdot QP \leq &\Delta f_{\rm ac}^{m1,n1}(x,y) + \Delta f_{\rm ac}^{m2,n2}(x,y)\\ &\leq 1.0\cdot QP \quad \left(\text{when } k = \frac{1}{2}\right). \qquad (10)\end{aligned}$$

Obviously, dc embedding causes less of a gray-level change than ac embedding provided that QP is larger than 2. Besides, dc embedding causes a constant gray-level shift (i.e., $\delta_{\rm dc}$) in a block and a maximal difference of three gray-levels between two adjacent blocks, which is often invisible to human perception. On the other hand, ac embedding may introduce ripple-like noise of larger amplitude $(1.0\cdot QP)$. To compare the PSNRs of I-frames after data embedding at different positions, six video sequences ("Akiyo," "BBC," "Flower," "Foreman," "Miss_Am," and "Salesman") are tested. It is found that dc embedding results in a less quality degradation (on average 0.18 dB) than ac embedding (on average 1.82 and 1.88 dB for

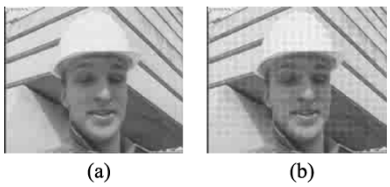

Fig. 1. Result of reconstructed I-frames ($QP = 13$) with information embedded at (a) 2 LSBs of dc coefficients by scheme 1 and (b) the first and second ac coefficients by scheme 2. Results of reconstructed I-frames with information embedded at the third and fifth ac coefficients by scheme 2 is similar to (b).

pairs of (first and second) and (third and fifth) ac coefficients, respectively). Fig. 1 shows the resulting image qualities. It is clear to find out that scheme 2 [Fig. 1(b)] causes distinguishable blocking artifacts than scheme 1 [Fig. 1(a)].

The major challenge of embedding information in P-frames comes from the large amount of skipped MBs and insufficient nonzero blocks in inter-coded MBs, especially for very low bit-rate coding. To be consistent, we propose that information can only be embedded in a nonskipped MB (including inter- and intra-coded) which contains at least one nonzero block. Depending on the number (1–6) of nonzero blocks in the host MB, information bits (MB_HL and/or MB_DL_8) are conditionally embedded according to the following four categorized cases.

- *Case 1*: For only one nonzero block, the 4-bit MB_HL information is totally embedded there.
- *Case 2:* For two nonzero blocks, the 4-bit MB_HL information is evenly embedded in them.
- *Case 3*: For three nonzero blocks, 2, 1, and 1 bits of MB_HL are sequentially embedded in them.
- *Case 4:* For more than three nonzero blocks, all information bits (4-bit MB_HL or 12-bit MB_HL || MB_DL_8, where "||" means concatenation) are evenly embedded in the first four (in the order of Y, Cb, and Cr) or all six nonzero blocks.

For each inter-coded MB to be modified, information bits are sequentially embedded into the first few low frequency (in the zigzag scanning order) DCT coefficients, regardless of zeros or not. After applying scheme 2 to a host MB, a resulting zero block should be prohibited to make data extraction possible. In this case, the DCT coefficient immediately after the last embedding position should be forcedly set to a nonzero value (e.g., 1 or -11). Furthermore, for the intra-coded MB_*before* (mostly case 4), the embedded information (4-bit or 12-bit MB_HL || MB_DL_8) is evenly embedded into the dc coefficients of all six blocks (i.e., 1 bit per Y-component block or 2 bits per block) by using scheme 1.

In case that the inter-coded MB_*before* contains only MV and no DCT blocks, our proposed scheme embeds no information for error resilience and thus introduces no quality degradation or bit-rate increase. In this case, the proposed scheme degenerates to the original H.263+ codec.

In general, case 1 will cause significant distortion for those MBs with a single nonzero block, whereas cases 2–4 are featured of distributing these distortions in more blocks. Case 1 also causes modifications of higher frequency DCT coefficients which may lead to a significant increase in coding bit rate.

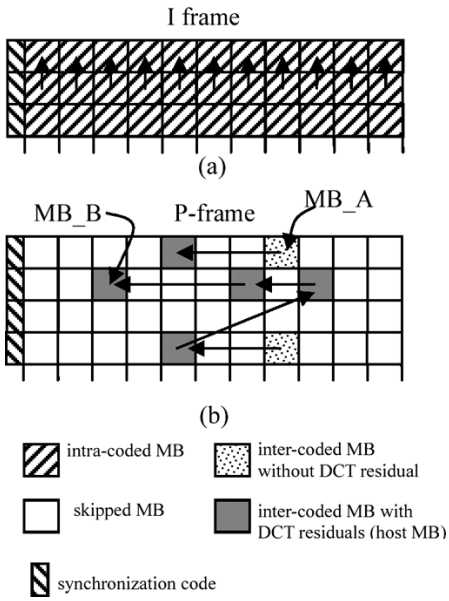

Fig. 2. Selection of host MBs for information embedding, as indicated by arrows. (a) I-frame: the upper-adjacent MB. (b) P-frame: the preceding nonskipped MB.

III. Proposed Error-Resilient Video Coding Algorithm

A. Information Embedding for I-Frames

Here, we propose to embed the MB_DL information of each MB in I-frames into the upper-adjacent MB (i.e., as the host MB), as shown in Fig. 2(a), by using scheme 1 described in Section II. The encoder has to temporarily store the kth GOB coded data in an extra memory buffer and then modify its dc coefficient part (fixed length codes) until the $(k+1)$th GOB is encoded. Note that this modification does not alter the data length of the kth GOB, hence the bit stream of the $(k-1)$th GOB can remain unchanged. In this manner, the encoding procedure only has a delay of one GOB. As for the first GOB in each I-frame, the embedding process is skipped due to the lack of host MBs.

B. Information Embedding for P-Frames

For P-frames, if errors occur in a run of skipped MBs, the immediately following nonskipped MB will not be decodable. To protect these errors, it requires that the run-length of the skipped MBs be recorded and embedded so that the decoder is capable of re-synchronizing the start of the following nonskipped MB. Hence, the host MB is no longer located at a fixed position just as in I-frames, but should be adaptively selected. The encoder has to temporarily keep the data of a MB until its corresponding MB_*after* is encountered and then modify the selected coefficients by schemes 1 or 2. However, embedding of MB_DL_8 of the MB_*after* in MB_*before* normally causes bit-rate variation, which subsequently changes MB_DL_8 of MB_*before*. This incurs the requirement of a backward procedure that processes the last MB first and concatenates the resulting bit streams in the forward direction. As a consequence, a one-frame delay is resulted, in contrast to the one GOB delay for I-frames. Fig. 2(b) shows how a nonskipped MB chooses its host MB (the MB pointed by an arrow). The exception occurs when the host MB contains

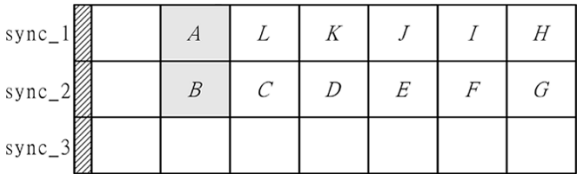

Fig. 3. Illustration of MB backward decoding when two vertically adjacent MBs (e.g., B and A) are both corrupted.

only one MV (or 4 MVs) but no DCT residuals [e.g., MB_A in Fig. 2(b)]. In this case, the embedding process is ignored.

C. Information Extraction for MB Resynchronization

The video decoding procedure is similar to the traditional H.263+, but with additional extraction of the embedded information for MB resynchronization. For each MB in I-frames, the extraction of embedded information is extremely simple: just capture the two LSB bits directly from each dc coefficient, replace them with a fixed "01" pattern (as mentioned in Section II-A), and then concatenate the extracted 12 bits to form the MB_DL information. Basically, the usage of MB_DL datum can be two-fold: 1) validating a decodable MB (i.e., implicit-error detection) and 2) skipping an error MB (i.e., error protection). Fig. 3 illustrates a situation that both a MB (B) and its host MB (A) incur errors. In this case, resynchronization of MB_C becomes impossible. Our decoder will immediately look for the next GOB startcode (sync_3) and put the bits there between into the memory buffer. At this newly found synchronization point, the decoder can process the bit stream backward by delimiting MBs_G, F, E, D, and C based on the extracted information from their host MBs (i.e., MBs_H, I, J, K, and L, respectively). This backward procedure stops until a corrupted host MB (e.g., MB_A) is encountered. Hence, even in the condition that two vertically adjacent MBs are both corrupted, our algorithm can still work well. However, there would have troubles when both the considered MB and its host are attacked by errors, but still decodable. Surely, this should be resorted to error detection technique. By applying the forward and backward decoding procedures with information extraction for each slice or GOB, the number of lost MBs (defined as those are undecodable due to error propagation) can be reduced to a minimal.

For P-frames, the selected ac coefficients in each nonskipped MB (as defined in cases 1–4 in Section II) are examined to obtain their modulo 2 values (i.e., 0 or 1). These extracted bits are concatenated to get the MB_HL information of following nonskipped MB. According to this information, the decoder passes by consecutive skipped MBs until reaching the nonskipped MB whose location (in a GOB row) is specified by the extracted information. If at this location a "0" bit is encountered, it is recognized as the start of a nonskipped MB. Otherwise (a "1" bit is encountered), the decoder continues to pass by n multiples of N "1's" ($N = 11$ for QCIF format), as well as the required resynchronization codes there between, until a "0" bit is met at the specified MB_HL location. Our system cannot definitely predict the value of n. This indefiniteness disables the protection of errors occurring at "1" bit which indicates a skipped MB

Error skipping process for I-frames

```
Decode MBi;
If (MBi is undecodable )
{
    Find the host MBi*;
    If ( MBi* is reconstructed )
      Extract MB_DL from DC component of MBi* and skip
      MB_DL bits;
    Else
    {
      Put the following bits in the buffer and search next
        synchronization code sync (assume MBj is located
        before the sync);
1:    Extract MB_DL from host MB of MBj;
      Localize the bit stream part of MBj in the buffer by the
        extracted MB_DL above;
      Decode MBj;
      j=j-1;
      Got to 1 for backwards localizing and decoding the MB bit
        stream until an error host MB is met;
    }
}
Decode next MB.
```

Error skipping processing for P-frames

```
Decode MBi;
If (MBi is non-skipped)
{
   Decode MBi and extract MB_HL and/or MB_DL_8;
   Check errors by counting following "1" (skipped MBs) until a
     non-skipped MBj (start with "0") is met at position MB_HL;
   Decode MBj;
   If (MBj is undecodable) skip MBj by MB_DL_8 bits (for
     Mod_IP4E" mode);
}
Decode next MB.
```

Fig. 4. Pseudo codes of error-skipping process for I- and P-frames.

at the same specified MB_HL location. For errors occurring in the bit stream of a nonskipped MB, the exacted MB_DL_8 information can be used to prevent error propagation. In case that the MB_DL_8 information is missing due to an undecodable host MB (i.e., these two nonskipped MBs are both in errors), the subsequent skipped and nonskipped MBs will be lost until next synchronization code is encountered. In this case (often in high error environments), the performance against errors will be the same as the traditional H.263+ codec.

A summary of the above-mentioned error-skipping algorithms for I- and P-frames are given in Fig. 4 for more clarity.

IV. Simulations

The experiments were preformed on eight QCIF 4:2:0 color video sequences coded at 10 frames per second (fps). Each group of pictures (GOP) is structured as IPPP… and contains 21, 51, or 101 frames. The quantization scale factor QP is set to 9–31 for medium-to-low bit-rate purpose. The GOB start-codes are inserted at the front of each MB row for the original H.263+ and proposed codecs. The following algorithms are compared.

1. Ori: Original H.263+ TMN8 codec with a fixed quantization scale factor.

2. ARMP: H.263+ codec enhanced with adaptive resynchronization marker position (ARMP) **[13]** for all frames.
3. Mod_I: Proposed method with data embedding in I-frames only.
4. Mod_IP1: Proposed method with simultaneous embedding of 12 bist in I-frames and 4-bit MB_HL in P-frames, where each eligible host MB should have at least one nonzero block.
5. Mod_IP2: Similar to Mod_IP1, except that each eligible host MB should have at least two nonzero blocks.
6. Mod_IP4: Similar to Mod_IP1, except that each eligible host MB should have at least four nonzero blocks.
7. Mod_IP4E: Proposed method with simultaneous embedding of 12-bit MB_DL in I-frames and 12-bit "MB_HL || MB_DL_8" in P-frames, where each eligible host MB should have at least four nonzero blocks.

Among them, Mod_IP1 gives chances to embed information even only one nonzero 8×8 block (Y, C_b, or C_r component) is available in a MB. On the other hand, "Mod_IP4" and "Mod_IP4E" strongly requires the existence of at least four nonzero 8×8 blocks for distributive embedding.

Under the same quantization step size, "Mod_I" and "Mod_IP4" produce a bit stream with almost the same size as H.263+. On the other hand, "Mod_IP1", "Mod_IP2", and "Mod_IP4E" increase the bit stream size by a perceivable amount. The induced extra bit rate, with respect to the original H.263+ codec, is figured out. The same extra amount of bit rate is then added to the H.263+ bit stream by inserting additional resynchronization codes (32 bits per code in H.263+) in all frames by using the ARMP algorithm [13]. This experiment was mainly designed to compare error resilience when the resynchronization information is embedded in the bit stream or conveyed in an explicit manner (i.e., ARMP).

To compare all schemes, the average PSNR is calculated as [17]

$$\mathrm{PSNR}_{\mathrm{avg}} = \frac{(4 \times \mathrm{PSNR}_Y + \mathrm{PSNR}_{Cb} + \mathrm{PSNR}_{Cr})}{6}. \quad (11)$$

To be fair in comparisons, error patterns are generated in the following procedures.

1) Error probability of each MB (including the skipped and nonskipped) in I- and P-frames is obtained by computing $1-(1-p)^n$, where p and n are bit error rate (BER) and number of bits in the considered MB, respectively. The same simulation manner was adopted for all seven test algorithms.
2) All synchronization codes in I- and P-frames were assumed error-free for simplicity.
3) Fifty runs of experiments were conducted.

To see tradeoffs between the hidden information and the increased bit rate due to data embedding, we define a Bnf (abbreviation of *Benefit*) as

$$Bnf = \mathrm{Bits}_{\mathrm{info}} + \mathrm{Bits}_{\mathrm{original}} - \mathrm{Bits}_{\mathrm{proposed}} \quad (12)$$

where the first two items represent the demanded bit rate when the embedded information is transmitted separately via another secure and synchronized channel; however, the third item $\mathrm{Bits}_{\mathrm{proposed}}$ stands for the created bit rate when the same amount of information is embedded for transmission. Bnf describes the number of bits that we can save via the proposed data embedding techniques. From our experimental results (not shown here due to limited space), two points can be found.

TABLE I
PSNR PERFORMANCES (IN dB) OF THE ORIGINAL H.263+ AND THE PROPOSED SCHEMES ("MOD_I" AND "MOD_IP4") WITH $\mathrm{GOP\_size} = 51$, $QP = 15$, AND DIFFERENT BERS

	Akiyo			BBC			Flower			Foreman		
BER	Ori	I	IP4	Ori	I	IP4	Ori	I	IP4	Ori	I	IP4
1.E-06	33.4	33.2	33.2	28.9	28.9	28.7	28.7	28.7	28.6	32.2	32.1	32.0
1.E-05	33.4	33.2	33.2	27.7	27.7	27.6	28.7	28.6	28.6	32.0	31.9	31.9
1.E-04	33.3	33.1	33.1	20.6	**21.0**	**21.1**	28.2	**28.3**	**28.3**	30.1	**30.3**	**30.2**
1.E-03	32.1	**32.7**	**32.7**	10.3	**11.3**	**11.4**	25.5	**26.2**	**26.2**	21.4	**23.2**	**23.3**
1.E-02	28.7	**29.1**	**29.1**	13.1	**13.2**	**13.2**	21.5	**21.7**	**21.6**	15.9	**16.2**	**16.0**
	Glasgow			Miss_Am			Salesman			Table Tennis		
BER	Ori	I	IP4	Ori	I	IP4	Ori	I	IP4	Ori	I	IP4
1.E-06	30.7	30.5	30.2	35.5	35.3	35.3	31.9	31.7	31.7	30.5	**30.7**	**30.6**
1.E-05	30.2	30.1	29.8	35.5	35.3	35.3	31.9	31.7	31.7	30.1	**30.3**	**30.3**
1.E-04	26.9	**27.2**	**27.0**	35.3	35.1	35.1	31.7	31.6	31.6	28.0	**28.4**	**28.3**
1.E-03	18.1	**19.5**	**19.4**	33.3	**33.6**	**33.6**	29.9	**30.6**	**30.5**	19.5	**21.6**	**21.5**
1.E-02	14.8	**15.1**	**15.1**	28.3	27.5	27.4	26.5	**26.6**	**26.6**	16.6	**17.3**	**17.3**

1) "Mod_I" and "Mod_IP4" are economical ($Bnf > 0$) for most of the QP values. The value of Bnf is nearly independent of QP for "Mod_I", since the QP for dc components is fixed at 8. For coding schemes that have a positive Bnf, a positive PSNR gain might be resulted even at BER $= 1 \times 10^{-4}$ (see Table I).
2) "Mod_IP4E", "Mod_IP1", and "Mod_IP2" are uneconomical ($Bnf < 0$). This is due to the fact that they cause more distortions in P-frames and thus increase the overall bit rate. In spite of this lose of economy, the excess bit rate (i.e., Bnf) may result in an improvement of error resilience in presence of channel errors. Please see Table II for this comparison.

Though "Mod_I" and "Mod_IP4" are more economical in terms of embedding benefit, their embedding capacity is actually limited. Accordingly, they present less error resilience in face of channel errors.

The rate-distortion performances in error-free condition ($\mathrm{GOP\_size} = 51$) were also compared. Since data embedding results in quality degradation and increasing residuals, the overall bit rate will be increased accordingly. Our rate-distortion curves are obtained by changing QP from 9 to 31. Four out of the eight cases are illustrated in Fig. 5. It was found from these 8 test sequences that "Mod_I" and "Mod_IP4" degrade the average PSNR by 0.06–0.3 dB and 0.13–0.3 dB, respectively, with respect to the original H.263+ codec (at the same bit rate). "Mod_IP4E" degrades the average PSNR by 0.4–1.8 dB, with respect to "ARMP" (at the same QP and bit rate). The nearly indistinguishable curves imply that "Mod_I" and "Mod_IP4" achieve almost the same visual quality as the original H.263+ codec in the error-free case. Similar performances can also be found in other test sequences with $\mathrm{GOP\_size} = 21$ and 101 (not shown here).

The exception comes from the "Akiyo" and "Miss America" sequences, where "Mod_I" and "Mod_IP4" present a larger PSNR degradation (≈ 0.3 dB) than in other sequences. This is mainly due to the simple and dark background content, which is susceptible to disturbances in dc components, as well as the nonlinear relation between PSNR and MSE (the same amount

TABLE II
PSNR PERFORMANCES (IN dB) OF ARMP AND PROPOSED SCHEMES ("MOD_IP1," "MOD_IP2," AND "MOD_IP4E") WITH GOP_size $= 51$, $QP = 9$, AND DIFFERENT BERS

	Akiyo				BBC				Flower				Foreman			
BER	ARMP	IP1	IP2	IP4E	ARMP	IP1	IP2	IP4E	ARMP	IP1	IP2	IP4E	ARMP	IP1	IP2	IP4E
1.E-06	36.1	36.1	35.7	35.1	31.3	31.3	31.3	30.5	31.8	31.0	31.1	30.6	34.4	33.6	33.8	32.7
1.E-05	36.1	36.1	35.7	35.1	26.9	**28.6**	**28.6**	**29.8**	31.5	30.8	30.9	30.6	33.6	33.1	33.3	32.6
1.E-04	36.0	35.8	35.6	35.0	15.8	**18.8**	**18.8**	**25.3**	30.2	29.5	29.9	30.0	28.9	**29.7**	**29.8**	**31.5**
1.E-03	35.2	33.1	34.2	33.6	11.2	10.7	10.7	**11.7**	27.5	23.9	25.1	25.7	20.8	20.5	20.7	**24.1**
1.E-02	29.7	28.8	28.9	27.1	10.2	**13.8**	**13.8**	**13.6**	23.1	20.2	21.4	19.7	15.8	**16.6**	**16.5**	**15.9**
	Glasgow				Miss_Am				Salesman				Table Tennis			
BER	ARMP	IP1	IP2	IP4E	ARMP	IP1	IP2	IP4E	ARMP	IP1	IP2	IP4E	ARMP	IP1	IP2	IP4E
1.E-06	33.1	32.5	32.7	31.8	37.4	36.0	36.6	35.5	34.2	33.7	33.8	33.3	33.4	32.7	32.9	32.3
1.E-05	31.8	31.4	31.5	31.5	37.3	35.9	36.6	35.5	34.2	33.6	33.7	33.3	32.6	31.8	32.0	32.0
1.E-04	25.9	**26.2**	**26.3**	**29.2**	36.9	35.2	36.1	35.2	33.7	33.1	33.3	33.1	30.1	28.0	28.2	29.9
1.E-03	18.6	17.8	17.9	**19.7**	33.9	31.2	32.8	33.2	30.7	30.0	30.3	30.7	22.8	19.9	19.9	20.9
1.E-02	14.3	**15.3**	**15.6**	**15.3**	28.2	24.4	26.0	24.7	24.4	**26.0**	**26.4**	**25.1**	16.4	**17.2**	**17.3**	**16.9**

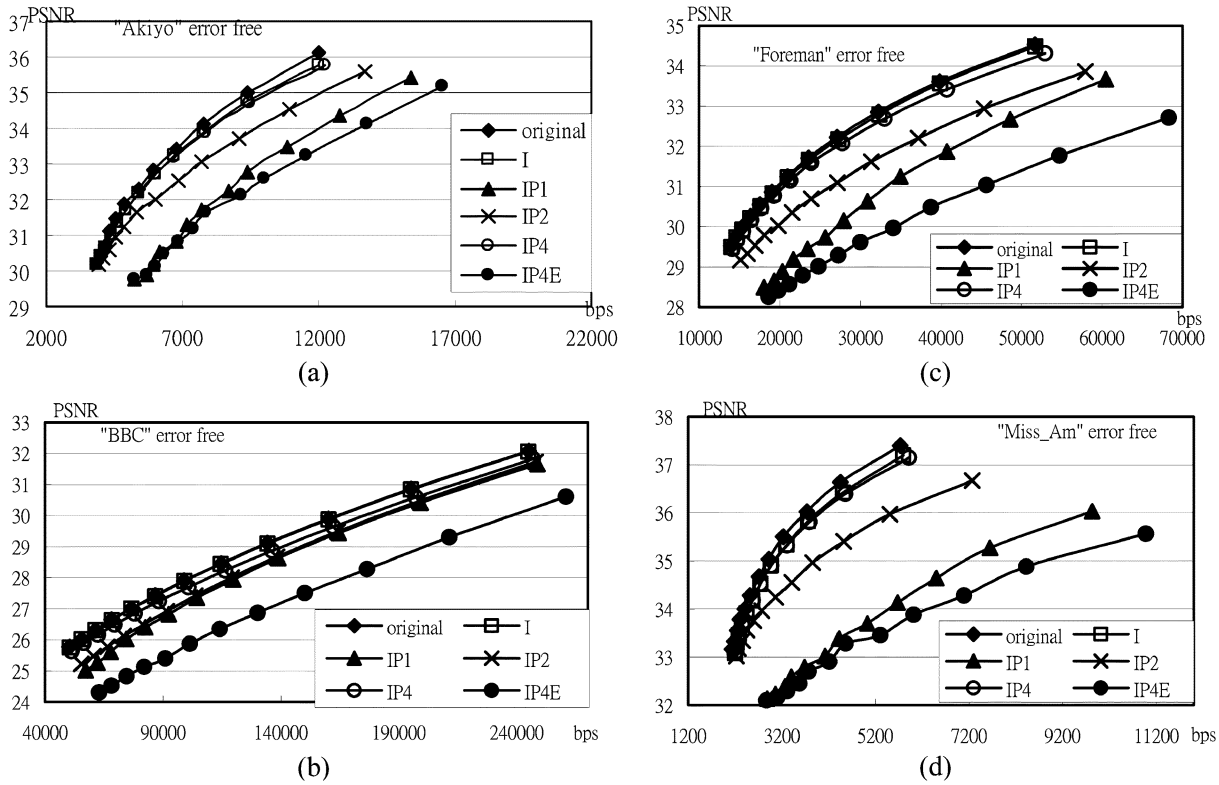

Fig. 5. Comparisons of rate-distortion performances between H.263+ and proposed schemes with GOP_size $= 51$ in error-free situation. The test sequences are (a) "Akiyo," (b) "BBC," (c) "Foreman," and (d) "Miss America."

of variation in MSE would result in a larger reduction in PSNR for higher quality sequences).

To evaluate the error resilience performance, each algorithm is integrated with the same error concealment technique, e.g., the zero-MV replacement method. Essentially, zero-MV replacement cannot be applied to the first frame of the video, which is then assumed to be error-free. Since "Ori," "Mod_I," and "Mod_IP4" result in a similar bit rate, we compare them in Table I. On the other hand, "Mod_IP1," "Mod_IP2," and "Mod_IP4E," which cause significant overheads, are compared in Table II with "ARMP." To be fair, the insertion of resynchronization codes in "ARMP" is such that the resulting bit rate is approximately the same as that of "Mod_IP4E." Both experiments were obtained with GOP_size $= 51$ and $QP =$ 9, 15, and 25. The choice of different QPs leads to different numbers of nonskipped MBs for information embedding. Due to limit in space, only a part of results is shown here (Table I is with $QP = 15$ and Table II with $QP = 9$). In Tables I and II, the boldfaced values mean that the proposed algorithms outperform "Ori" or "ARMP" in PSNR. From Tables I and II, several key points can be found below.

1) For all test sequences, "Mod_I" and "Mod_IP4" mostly provide better error resilience capability than "Ori," as BER becomes larger than 1×10^{-4}. For examples, there are 1.9 and 2.0 dB of improvement for "Foreman" and "Table_tennis," respectively, when BER $= 10^{-3}$.
2) Though "Mod_IP1," "Mod_IP2," and "Mod_IP4E" sacrifice Bnf in view of the embedding cost, they might provide better error resilience than "ARMP," as long as properly applied. For example, for videos of

TABLE III
COMPARISON BETWEEN THE PROPOSED ALGORITHMS AND OTHER DATA-EMBEDDING-BASED METHODS
(BER: BIT ERROR RATE, BKER: BLOCK ERROR RATE, GOBLR: GOB LOSS RATE)

Method	[8]	[9]	[10]	[11]	[12]	Proposed
Target/ Standard	Video/ H.263	Video/ H.263	Image/ JPEG	Image/ JPEG	Video/ MPEG4	Video/ H.263+
Functionality	Error detection	Error detection	Error concealment	Error concealment	Error detection	Error skipping (resynchronization)
What to embed	Parity check bits	Parity check bits	Image features	VQ index	Parity check bits	MB_DL MB_HL
Where to embed	MVs	MVs, DCT coefficients	DCT coefficients	DCT coefficients	DCT coefficients	DCT coefficients
How to embed	Set to specific MV	Set to specific MV, modulo 2	modulo 2	modulo 2	modulo 2	LSB replacement, modulo 2
Quality degradation	0.3~1.0dB	0.5~1.5dB	0.32dB	about 1dB	0.1~0.27dB	0.13~0.3dB (Mod_I, Mod_IP4) 0.4~1.8dB (Mod_IP4E)
Gain /error rate (compared to original codec)	on average 1dB / GOBLR≤5%	up to 0.6dB /BER≤10^{-3}	0.37dB/ BKER= 25%	2~3dB/ BKER ≤ 30%	up to 0.5dB/ BER≤10^{-3}	up to 2dB/ BER≤10^{-3} (Mod_I, Mod_IP4) up to 9.5dB/ BER≤10^{-3} (Mod_IP4E)

dynamic content (such as "BBC") which are coded at high bit rate (i.e., low QP), the considerable amount of nonskipped MBs will make our algorithm effective in conveying error-skipping information. As seen from Table II, for "BBC," "Mod_IP1," "Mod_IP2," and "Mod_IP4E" mostly outperform "ARMP" by 0.5–9.5 dB when $QP = 9$ and BER $= 1 \times 10^{-5}$–1×10^{-2}.

3) For low-motion sequences such as "Akiyo" and "Miss America," which are normally coded at low bit rates, P-frames will contain a large portion of skipped MBs and nonskipped MBs with few nonzero blocks. In this case, "Mod_IP4" is a reasonable choice, considering the tradeoffs between bit rate increase and error resilience. For example, "Mod_IP4" outperforms the original H.263+ codec by 0.6 dB when BER equals 10^{-3} for "Akiyo." On the other hand, all the performances of "Mod_IP1," "Mod_IP2," and "Mod_IP4E" are worse than ARMP due to excessive embedding noise. Since for low-motion sequences, the zero-motion replacement technique for error concealment is sufficient to compensate the lose of a series of MBs, the benefit of data embedding become insignificant.
4) In order to provide more powerful error resilience capability, more information has to be embedded, but it will also cause more quality degradation in error-free condition. This would be a tradeoff between the (error-free) quality degradation and the error resilient capability. In other words, "when" to use data embedding is important. When channel condition is good, it is not smart to adopt the "Mod_IP4E" (heavy embedding) coding method. When the channel is bad, "Mod_I" or "Mod_IP4" (light embedding), though causing less degradation $(< 0.3$ dB$)$ in error-free condition, are not enough to resist the channel errors. In application of "Mod_IP4E," the video content and the value of QP would also be taken into consideration, since "Mod_IP4E" requires more nonskipped MBs for data embedding.

It is important to note that our experiments have considered the operation of error concealment (here, the zero-MV replacement). Hence for low-motion video (e.g., the "Akiyo" or "Miss America"), the loss of MBs can be almost compensated by error concealment. This is why the original H.263+ codec performs better than our proposed methods in some cases. This is also why the error resilience coding schemes that rely on the synchronization codes do not show the same performance in all kinds of video sequences.

In summary, we suggest the suitability of proposed schemes by considering channel BER and the amount of motion-compensated residuals (depending on the video content and QP values) as follows.

1) "Mod_I" is enough for channels of low BER and videos with low motion-compensated residuals.
2) "Mod_IP4E" which embeds more error resilience information works better for channels of high BER and videos with high motion-compensated residuals.
3) "Mod_IP4" which plays a tradeoff between "Mod_I" and "Mod_IP4E" is a better choice for other channel conditions.

V. CONCLUSION AND REMARKS

In this paper, error-resilient video coding schemes based on data embedding techniques were proposed for the H.263+ codec. It is in principle of establishing a covert channel to convey important information to the decoder end for error protection, without significant increase in transmission bit rate. The embedded information, here the data length of MBs in I-frames and horizontal locations of nonskipped MBs in P-frames, is capable of providing implicit MB delimiters for resynchronization in presence of channel errors. Various schemes, such as "Mod_I," "Mod_IP1," "Mod_IP2," "Mod_IP4," and "Mod_IP4E," are extensively analyzed and compared to some existing schemes (e.g., the original H.263+ TMN8 and the ARMP method). Experiments show that our data embedding

process decreases the average PSNR by less than 0.3 dB (light data embedding) and 1.8 dB (heavy data embedding) in error-free condition, but have a significant PSNR improvement of up to 2 and 9.5 dB, respectively, when BER varies between 10^{-5} and 10^{-3}. Besides, suitability of proposed schemes is figured out from experiments, which indicates that "Mod_I" is most suitable for low residual videos (i.e., high QP and static content) in low BER condition, "Mod_IP4E" is most suitable for high residual videos (i.e., low QP and dynamic content) in high BER condition, and "Mod_IP4" may be appropriate elsewhere. However, if there exists no host MBs in P-frames (e.g., at very low bit rate for low-motion video), the proposed scheme will degenerate to "Mod_I."

In Table III, several data-embedding-based error resilience/error concealment methods and our proposed algorithm are compared qualitatively. Quantitative performances of the compared methods, e.g., the quality degradation with data embedding and the gain in presence of channel errors, are referred from their published materials. Since some papers did not show enough experimental results (e.g., only one image or error condition was tested), only a specific value, instead of a range, is listed. Most of the proposals focus on error detection of video bit streams. Since the embedded information in this case is not critical, PSNR gain in presence of noises is also not significant. On the other hand, embedding VQ information for image error concealment though yields significant PSNR improvement, but relies on the availability of codebook at receiver side (surely, it also consumes some bit rates, if optimality of VQ is desired). We know from the above that data embedding techniques play tradeoffs between quality degradation (in error-free condition) and error resilience (in presence of heavy channel noise). It will be more efficient if the knowledge of "when" to apply this kind of techniques is known. That is, it will be better if a technique of channel estimation is provided. Even, the estimation of motion activity of a video would be helpful in choosing suitable coding scheme among "Mod_I," "Mod_IP4," and "Mod_IP4E." In summary, the 4W issues, "what," "how," "where," and "when" should be the main concerns in the application of data embedding techniques for error resilient visual communication.

REFERENCES

[1] M.-T. Sun and A. R. Reibman, *Compressed Video Over Network*. New York: Marcel Dekker, 2001.

[2] W. N. Lie and L. C. Chang, "Robust and high-quality time-domain audio watermarking based on low-frequency amplitude modulation," *IEEE Trans. Multimedia*, vol. 8, no. 1, pp. 46–59, Feb. 2006.

[3] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communication," *Proc. IEEE*, vol. 86, no. 5, pp. 974–997, May 1998.

[4] J. Wen and J. D. Villasenor, "Reversible variable length codes for efficient and robust image and video coding," in *Proc. Data Compression Conf.*, 1998, pp. 471–480.

[5] D. W. Redmill and N. G. Kingsbury, "The EREC: An error-resilient technique for coding variable-length block of data," *IEEE Trans. Image Process.*, vol. 5, no. 4, pp. 565–574, Apr. 1996.

[6] "Report of Results on Core Experiment on Error Resilience for Motion Data With Structured RVLC -E8,", ISO/IEC JTC1/SC29/WG11/M2382, 1997.

[7] A. H. Li, M. Fong, J. Wen, and J. D. Villasenor, "Test Results of Error Resilience With Modified Error Resilient Syntax With Data Partitioning and RVLC,", ITU-T Documentation Q15-E-20, 1998.

[8] J. Song and K. J. R. Liu, "A data embedded video coding scheme for error-prone channels," *IEEE Trans. Multimedia*, vol. 3, no. 6, pp. 415–423, Dec. 2001.

[9] "An Error Detection Scheme using Data Embedding for H.263 Compatible Video Coding,", ISO/IEC JTC1/SC29/WG11 MPEG99/N6340, 2000.

[10] P. Yin, B. Liu, and H. H. Yu, "Error concealment using data hiding," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2001, pp. 1453–1456.

[11] L. W. Kang and J. J. Leou, "An error resilient coding scheme for JPEG image transmission based on data embedding and side-match vector quantization," J. Vis. Commun. Image Representation, 2006, to be published.

[12] H. Okada, A. E. Shiitev, H. S. Song, G. Fujita, T. Onoye, and I. Shirakawa, "Error detection by digital watermarking for MPEG-4 video coding," *IEICE Trans. Fundamentals*, vol. E85-A, no. 6, pp. 1281–1288, Jun. 2002.

[13] K.-Y. Yoo, "Adaptive resynchronization marker positioning method for error resilient video transmission," *Electron. Lett.*, vol. 34, no. 22, pp. 2084–2085, 1998.

[14] "ITU-T Recommendation H.263 Version 2, Video Coding for Low Bit Rate Communication," International Telecommunication Union, Geneva, 1998.

[15] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding—A survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

[16] J. Huang, Y. Q. Shi, and Y. Shi, "Embedding image watermarks in dc components," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 6, pp. 974–979, Sep. 2000.

[17] H. C. Shyu and J. J. Leou, "Detection and concealment of transmission errors in MPEG images-a genetic algorithm approach," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 6, pp. 937–948, Sep. 1999.

[18] W. N. Lie and G. S. Lin, "A feature-based classification technique for blind image steganalysis," *IEEE Trans. Multimedia*, vol. 7, no. 6, pp. 1007–1020, Dec. 2005.